Optimizing Feature Weighting and Project Selection for Analogy Based Estimation of Software Development Effort

(Draft 1.0)

William Sica
West Virginia University
Morgantown, West Virginia
Email: w.t.Sica@gmail.com

***Abstract*—Estimating the effort required for software development is crucial for project management. Analogy based estimation (ABE) is a commonly used method for effort estimation which compares the new project to similar historical projects. ABE is criticized for low accuracy, and pruning the historical dataset has been suggested to improve accuracy. Feature weighting (FW) weights components of the project data, and project selection (PS) reduces historical data to representative projects. Optimizing FW and PS for ABE (FWPSABE) has been proposed in previous works. In this paper, we continue a previous study on FWPSABE and more extensively verify its utility in increasing estimation accuracy.**

I. Introduction

Software effort estimation is a nescesarry component of proper software project management. It effects many phases of management, from planning to budgeting. Improper estimates can cause infeasible deadlines and overbudget projects that may result in project cancellation and wasted funds.

Because of the importance of software effort estimation, many methods have been proposed to create accurate estimates.

A frequently used method for effort estimation is analogy based estimation (ABE). ABE finds analogous projects in a historical dataset, and estimates effort based on the effort required for these similar historical projects. While ABE has a widespread use, it is criticized for low prediction accuracy. As such, there is a demand for increasing the accuracy of ABE, and many methods have been proposed.

We focus particularly on the methods of feature weighting (FW) and project selection (PS) to improve the accuracy of ABE. Feature weighting supposes that certain features of a project have more impact on the effort required to complete it, and weights features relative to their impact. Project selection creates a representative sample of the historical data, in an effort to remove outliers.

In Section II, we give a more detailed explanation of FWPSABE.
In Section III, an explanation of optimization technique used is given.
In Section IV, the details of how the experiments were carried out is given.
In Section V, the results are displayed.
In Section VI, we restate conclusions and propose avenues for future work.

II. Background

A. Analogy Based Estimation (ABE)

Analogy based estimation is a form of case based reasoning which estimates a desired feature based on the value of that feature in the historical dataset. ABE works as follows

1) Gather relevant features from the new project.
2) Find similar projects in the historical database using a similarity function. (eg the k-nearest neighbors using Euclidean distance)
3) Predict the target feature from its value in the similar projects.

Different similarity functions and prediction methods exist for ABE. In this paper, ABE is performed using the following criteria:

5-nearest neighbors using Euclidean distance
Median of the effort feature of the 5-nearest neighbors

The Euclidean distance between two features is defined as follows:

For numeric featutes:		$d = (f_1 - f_2)$
For discrete features:	if $f_1 = f_2$	$d = 0$
	if $f_1 \neq f_2$	$d = 0$

Where f_1 and f_2 are features in the projects whose distance is being computed. The euclidean distance between projects is the sum of the Euclidean distance between all component features. The nearest-neighbors are the projects in the historical dataset with the lowest Euclidean distance to the new project.

The median is used for effort calculations because outliers with high or low effort can skew the mean dissproportionately for a small sample of 5 values.

B. Feature Weighting and Project Selection (FW and PS)

Feature weighting multiplies the distance value of features by a specified weight. In this way, features with a higher weight will have a greater impact on the distance between two features. This is useful when certain features have more impact on the feature to be estimated than others.

Project selection prunes the historical database to a smaller subset of representative examples. This has positive impacts

on estimation accuracy when examples exist in the historical database which are excessive outliers or disproportionately skewed relative to the rest of the data.

C. Magnitude of Relative Error

In order to evaluate the accuracy of an estimate, the magnitude of relative error is used. Given an estimated effort E and the actual effort of the project A, the magnitude of relative error is

MRE = $\left|\frac{A-E}{A}\right|$

Further, the mean magnitude of relative error (MMRE) is the mean of the MRE values evaluated in the test space. The median magnitude of relative error (MdMRE) is the median of the MRE values evaluated in the test space.

III. Description of Algorithm

Different guided-search optimization techniques were applied to the domain, but there was not a significant difference between the optimization techniques. As such, a genetic algorithm based on Li's work was used such that the algorithm's execution could be examined at a particular generation and to compare results gathered to previous work.

Optimization requires a fitness function with which to evaluate the relative value of results. The optimization methods given below assume the feature-weights and project selections are used to evaluate a set of training data given a historical dataset. The MMRE of the training data is given as the fitness, and the optimization seeks to find the set of feature-weights and project selection which produces the minimum MMRE. Optimizing on MdMRE was attempted, but while the training MdMRE decreased it had no impact on the testing MdMRE. Optimization of feature weighting and project selection is done by evaluating the fitness of bit-strings representing feature weights and selected projects. The bit-string used is divided in to two parts, one representing the feature weights and two representing the projects selected.

Part one stores two-bits for each feature, which corresponds to a weighting. The weights used are given:

$BitValues$	$Weight$
00	0
01	1
10	2
11	4

The weights are exponential so that significant weight differences can exist, but use only two-bits to reduce the length of the bit-string.

Part two of the bit string is the length of the number of projects, and stores a value of '1' when the project is included for consideration and '0' when it is excluded.

Given a number of projects m each with a number of features n (including effort), each bit-string is of length $m + 2n$.

A. Genetic Algorithm

This paper uses a genetic algorithm with the following procedure:

1. Create an initial population.

The function creates 10 bit-strings of size $m + 2n$, with each bit being assigned a 1 or 0 with probability .5.

2. Evaluate the population

The popuation is evaluated, by finding the MMRE of the bit-string on the training data. The fitness is the reciprocal of the MMRE, such that higher fitness values represent lower MMRE values. Maximum fitness value is sought.

3. Elitist Strategy

If the best fitness from the current generation is less than the overall best, the best bit-string of this generation is replaced by the best bit-string overall.

4. Select Bit-Strings to Mate

Roulette wheel selection is used to choose bit strings for mating. The cumulative fitness of the population is found, given as the sum of all fitness values in the current population. Each bit-string is then given a probability for selection, equal to its fitness over the cumulative fitness. Bit-strings are randomly chosen until a population for mating has been generated.

5. Crossover

Bit-strings selected for mating perform crossover with one another. At any particular bit along the bit-string, there is probability .7 of switching between the bit-strings. Such that, given bit-strings

000

111

A result of crossover might be the bit-strings

101

010

6. Mutation

Bits are randomly flipped along the populations bit-strings with probability .1. This is done to prevent the population from converging on local optima.

7. Proceed to Next Generation

The algorithm is run for 100 generations, starting at Step 2 after the first generation.

IV. Methodology

A. Preparing Training and Testing Data

The datasets were prepared as follows:

1. Remove Discrete Attributes for Numeric Features

The Desharnais dataset had the discrete attribute "?" for some projects as a numeric feature. Projects with this combination were removed from the dataset.

2. Normalize Features

Except for the target-feature of effort, other features were normalized in the range 0 to 1.

3. Randomly Permutate the Data

The datasets were randomly shuffled with 2/3rds being divided in to the Training Dataset and 1/3rd being divided in to the Testing Dataset. This was done 20 times for each dataset, so that 20 permutations existed for testing.

B. Information on Datasets Used

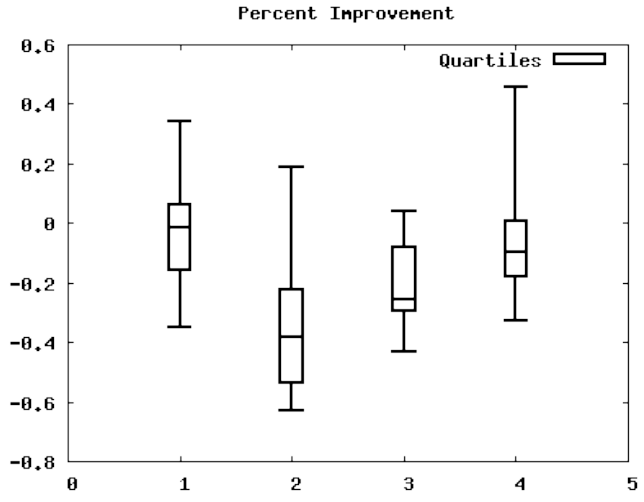
Dataset	Features	Projects
Albrecht	7	24
Cocomo '81	17	63
Desharnais	12	77
Maxwell	27	62

C. Trials

Each permutated dataset was run for four trials for each dataset. The percent improvement was calculated as:
$\frac{FWPSABETestingMMRE - ABETestingMRE}{ABETestingMMRE}$
This value was averaged over the four trials. Negative values denote a decrease in MMRE, and thus an increase n accuracy.

V. Results

This needs to be horizontal + labled, which I assume is done in LaTeX. I am not sure how to do that though.



]

Fig. 1. Column 1: Albrehct Dataset Column 2: Cocomo '81 Dataset Column 3: Desharnais Datase Column 4: Maxwell Dataset

VI. Conclusion & Future Works

Insufficient current data to make a conclusion. Data needs to be obtained for: Nasa '93 All 5 datasets using FW alone All 5 datasets using PS alone

As for 'How long does this take', 20,000 calls to FW-PSABE/FWABE/PSABE/ABE per test, with calls taking time based on dataset size. 3 hr / test 44 tests. Given 1 Lab Compy = 11 tests per processor, 33 hrs

Have been doing my tests previously by setting computers on for weekend and leaving them running.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.